(19)日本国特許庁（ＪＰ）　(12)特　許　公　報（Ｂ２）　(11)特許番号

第2923002号

(45)発行日　平成11年(1999)７月26日　(24)登録日　平成11年(1999)４月30日

(51)Int.Cl.[6]		識別記号	FI		
G09G	5/22	620	G09G	5/22	620
	5/24	670		5/24	670

請求項の数５(全 8 頁)

(21)出願番号　特願平2－213748

(22)出願日　平成２年(1990)８月14日

(65)公開番号　特開平4－96095

(43)公開日　平成４年(1992)３月27日

審査請求日　平成７年(1995)２月24日

(73)特許権者　999999999
沖電気工業株式会社
東京都港区虎ノ門１丁目７番12号

(72)発明者　大宅　伊久雄
東京都港区虎ノ門１丁目７番12号　沖電気工業株式会社内

(72)発明者　守谷　信行
東京都港区虎ノ門１丁目７番12号　沖電気工業株式会社内

(72)発明者　畠中　啓
東京都港区虎ノ門１丁目７番12号　沖電気工業株式会社内

(74)代理人　弁理士　香取　孝雄

審査官　大野　弘

最終頁に続く

(54)【発明の名称】　フォントフリーなラスタイメージ処理システム

(57)【特許請求の範囲】

【請求項１】ページ記述言語で記述されたデータを処理してラスタイメージを形成するラスタイメージ処理システムにおいて、該システムは、

前記データを受け、該データに含まれる文字のフォントデータを得るためのフォントデータ指定情報を作成し、得られたフォントデータからラスタイメージを形成するラスタイメージ処理手段と、

文字フォントを表す輪郭データが各装置共通に使用できる形式で蓄積されているデータベースにフォントデータ指定情報に基づいてアクセスして、該フォントデータ指定情報に関連する輪郭データを取り込むフォント管理手段と、

前記ラスタイメージ処理手段およびフォント管理手段の間をフォントに依存しない共通の形式でインタフェースするインタフェース手段とを含み、

前記フォント管理手段は、前記輪郭データを一時記憶する第１の蓄積手段を有し、

該フォント管理手段は、

前記フォントデータ指定情報に応じて、まず第１の蓄積手段にアクセスし、該フォントデータ指定情報に関連する輪郭データが第１の蓄積手段に存在すれば、これに基づいてフォントデータを生成し、

該フォントデータ指定情報に関連する輪郭データが第１の蓄積手段に存在しないときは、前記データベースに該フォントデータ指定情報に基づいてアクセスして、該フォントデータ指定情報に関連する輪郭データを取り込んでこれに基づいてフォントデータを生成するとともに、該取り込んだ輪郭データを第１の蓄積手段に蓄積することを特徴とするフォントフリーなラスタイメージ処理シ

ステム。

【請求項２】請求項１に記載のシステムにおいて、前記フォント管理手段は、

前記形成されたラスタイメージに含まれるラスタ化された文字データを一時記憶する第２の蓄積手段を有し、

前記ラスタイメージ処理手段がラスタイメージを形成する際、ラスタ化された文字データを第２の蓄積手段に蓄積し、

前記フォントデータ指定情報に応じて、第１の蓄積手段へのアクセスに先立って第２の蓄積手段にアクセスし、該フォントデータ指定情報に関連するラスタ化された文字データが第２の蓄積手段に存在すれば、これに基づいてフォントデータを生成し、

該フォントデータ指定情報に関連するラスタ化された文字データが第２の蓄積手段に存在しないときは、該フォントデータ指定情報に基づいて第１の蓄積手段にアクセスすることを特徴とするラスタイメージ処理システム。

【請求項３】請求項１に記載のシステムにおいて、第１の蓄積手段は、前記輪郭データのうち予め定められた相対的に使用頻度の高いものを常駐的に蓄積する第１の記憶領域と、相対的に使用頻度の低いものを非常駐的に蓄積する第２の記憶領域とを含むことを特徴とするラスタイメージ処理システム。

【請求項４】請求項１に記載のシステムにおいて、該システムは、前記輪郭データがフォントに依存しない共通の形式で蓄積され前記フォントデータ指定情報を受けるとそれに関連する輪郭データを出力するデータベース手段を含むことを特徴とするラスタイメージ処理システム。

【請求項５】請求項１に記載のシステムにおいて、該システムは、前記ページ記述言語で記述されたデータを作成して前記ラスタイメージ処理手段に供給するホストを含むことを特徴とするラスタイメージ処理システム。

【発明の詳細な説明】

（産業上の利用分野）

本発明はラスタイメージ処理システム、より具体的には、ページ記述言語で記述されたデータを処理してラスタイメージを形成するラスタイメージ処理システムに関するものである。

（従来の技術）

文字および画像データを含むコマンドデータは、たとえばパブリッシング用アプリケーションプログラムによって作成され、ページ記述言語で記述される中間ストリームに変換される。この中間ストリームは、ラスタイメージ処理（RIP）方式によるインタプリタによってラスタイメージに変換され、プリンタや表示装置などのイメージ出力装置にページの形で出力される。

文字を扱うシステムでは、様々な言語や国に応じて多様なフォントが存在し、したがってフォントに対しては多様な要求がある。たとえば、字母、書体、大きさのみならず、横書き、縦書き、左書き、右書き、斜字体、太字などの多様性がある。また、文字列すなわち文書としてフォントを扱う場合は、文字列の配列の仕方、たとえばプロポーショナル印字などの配列方式にも様々な要求がある。さらに、文字を特定するコードについては、フォントの供給者によって様々な文字コード系列が使用されている。

ラスタイメージ処理方式は、これらの多様な要求条件を少なくとも部分的に満足させねばならない。なお、本明細書において用語「フォント」は、単に字母や字体のみならず、上述のような様々な要求条件をも含む広義に解釈するものとする。

字母データは、フォント供給者に固有の形式、たとえばビットマップ形式やアウトライン（輪郭）形式などで提供される。フォントデータは一般に、文字の輪郭の直線および曲線を規定するディジタル化されたデータの形で外部ファイルやROMなどのメモリに蓄積される。ホストから特定の文字コードでフォントが指定されると、それに対応するフォントデータがファイルから読み出され、拡大または縮小などのスケーリング処理を施され、最終的にはラスタ化されてイメージに変換される。

ビットマップ形式やアウトライン形式による原データをラスタイメージに展開する処理は、フォントの形式によって相違することが多い。また、原データをスケーリングした後、長さや丸みを補正して適切な字形にするヒンティング処理も、フォントフォーマットによって異なることがある。

このようなパブリッシングの環境の下で従来は、フォントの種別に固有のラスタイメージ処理方式、すなわちフォント指向型の処理方式がとられていた。より詳細には、たとえば従来のプリンタは、ラスタイメージ処理系とフォントデータファイルを含み、セントロニクスやRS－232Cなどの汎用インタフェースでパソコンやワークステーションなどのホストに接続される。つまり、プリンタ内にあるラスタイメージ処理系とフォントデータは、特定のプリンタに固有であり、ホストはそれらを汎用のプリンタインタフェースを介して使用する構成がとられていた。また、CRTなどの画像表示装置は、ホストに接続されるが、表示装置で使用されるフォントのデータファイルと、それを扱うラスタイメージ処理系はホストの内部に含まれ、したがってホストに依存する構成がとられていた。

（発明が解決しようとする課題）

これからわかるように従来は、前述した様々な要求条件に適応的に対応できる単一のラスタイメージ処理システムが利用できなかった。より詳細には、特定のアプリケーションソフトが用いるコマンドデータには、それに固有のラスタイメージ処理系が必要であった。たとえば、PostScript（商標）などのページ記述言語によるコマンドデータストリームには、PostScriptインタプリタ

がラスタイメージ処理系として用いられる。つまり、特定のラスタイメージ処理系には、特定のフォント形式、フォント処理およびフォント読出し機能が組み込まれていた。

一般に、パブリッシングなどのドキュメント処理システムでは、複数のコマンドデータストリームが用いられる。したがって従来方式において、複数のデータストリームの扱いを可能にするためには、それぞれのコマンドストリームごとにフォントを用意しなければならなかった。これは、フォントの処理および管理システムの重複作成を招き、フォントの重複投資を誘起していた。

ところで、文字の表示を高速化するために従来は、一度使用されラスタ化された文字データをキャッシュに格納し、以降、同じ文字データを表示出力する際には、キャッシュからこれを読み出して使用するシステムがあった。これによって、複数回使用される文字データは、フォントデータをラスタ化する処理を行なわないで済む。

しかしこの従来方式では、キャッシュに蓄積されるラスタ化文字データは、サイズがその使用された状態に固定される。したがって、次に同じ文字フォントを異なったサイズで使用するときは、キャッシュに蓄積されている文字データを使用できず、別途、文字フォントデータをファイルよりフェッチしなければならなかった。仮りに、同じ文字フォントについて複数のサイズでラスタ化された文字データをキャッシュに蓄積するようにシステムを構成したとすると、当然ながら多くの記憶領域を必要とするであろう。この傾向は、漢字のように字種の多いフォントについては著しい。

本発明はこのような従来技術の欠点を解消し、高速な文字表示を行なうことができ、フォントに依存しないラスタイメージ処理システムを提供することを目的とする。

（課題を解決するための手段）

本発明によれば、ページ記述言語で記述されたデータを処理してラスタイメージを形成するラスタイメージ処理システムは、前記データを受け、このデータに含まれる文字のフォントデータを得るためのフォントデータ指定情報を作成し、得られたフォントデータからラスタイメージを形成するラスタイメージ処理手段と、文字フォントを表わす輪郭データが各装置共通に使用できる形式で蓄積されているデータベースにフォントデータ指定情報に基づいてアクセスして、フォントデータ指定情報に関連する輪郭データを取り込むフォント管理手段と、ラスタイメージ処理手段およびフォント管理手段の間をフォントに依存しない共通の形式でインタフェースするインタフェース手段とを含み、フォント管理手段は、輪郭データを一時記憶する第１の蓄積手段を有し、フォント管理手段は、フォントデータ指定情報に応じて、まず第１の蓄積手段にアクセスし、フォントデータ指定情報に関連する輪郭データが第１の蓄積手段に存在すれば、これに基づいてフォントデータを生成し、フォントデータ指定情報に関連する輪郭データが第１の蓄積手段に存在しないときは、データベースにフォントデータ指定情報に基づいてアクセスして、フォントデータ指定情報に関連する輪郭データを取り込んでこれに基づいてフォントデータを生成するとともに、取り込んだ輪郭データを第１の蓄積手段に蓄積する。

また本発明によれば、フォント管理手段は、形成されたラスタイメージに含まれるラスタ化された文字データを一時記憶する第２の蓄積手段を有し、ラスタイメージ処理手段がラスタイメージを形成する際、ラスタ化された文字データを第２の蓄積手段に蓄積し、フォントデータ指定情報に応じて、第１の蓄積手段へのアクセスに先立って第２の蓄積手段にアクセスし、フォントデータ指定情報に関連するラスタ化された文字データが第２の蓄積手段に存在すれば、これに基づいてフォントデータを生成し、フォントデータ指定情報に関連するラスタ化された文字データが第２の蓄積手段に存在しないときは、フォントデータ指定情報に基づいて第１の蓄積手段にアクセスする。

さらに本発明によれば、第１の蓄積手段は、輪郭データのうち予め定められた相対的に使用頻度の高いものを常駐的に蓄積する第１の記憶領域と、相対的に使用頻度の低いものを非常駐的に蓄積する第２の記憶領域とを含む。

（作用）

本発明によれば、ページ記述言語で記述されたデータがホストからラスタイメージ処理システムに入力されると、ラスタイメージ処理手段は、このデータに含まれる文字のフォントデータを得るためのフォントデータ指定情報を作成する。フォント管理手段は、フォントデータ指定情報に応じて、まず第１の蓄積手段にアクセスする。この輪郭データが第１の蓄積手段に存在すれば、これに基づいてフォントデータを生成する。しかし、存在しないときは、データベースにフォントデータ指定情報に基づいてアクセスして、そのフォントデータ指定情報に関連する輪郭データを取り込み、これに基づいてフォントデータを生成する。これとともに、取り込んだ輪郭データを第１の蓄積手段に蓄積する。

ラスタイメージ処理手段は、こうして得られた輪郭データからラスタイメージを形成する。形成されたラスタイメージは、最終的にはラスタイメージ出力装置や通信装置へ出力される。

（実施例）

次に添付図面を参照して本発明によるラスタイメージ処理システムの実施例を詳細に説明する。第１図を参照すると、この実施例は基本的には、たとえばパソコンやワークステーションなどの処理システムからなるホスト10と、ラスタイメージ処理（RIP）を行なうモジュールパッケージすなわちRIP処理モジュール12を含み、両者

がコネクタ14および16で相互に接続される。ホスト10は、文字および画像データを含むコマンドデータを、たとえばパブリッシング用アプリケーションプログラム50（第２図）によって作成し、ページ記述言語で記述される中間ストリームに変換する、本実施例では汎用の処理装置である。

ホスト10には、たとえばCRTなどの文字および画像を可視表示する画像表示装置18が接続されている。またホスト10および（または）RIP処理モジュール12にはプリンタ20が接続されている。プリンタ20は、文字および画像を記録媒体に可視記録する画像出力装置である。勿論、ホスト10は通常の処理システムのようにキーボードなどの入出力装置も含むが、これらは本発明の理解に直接関係ないので図示と説明を省略する。

本システムで特徴的なのは、表示装置18およびプリンタ20で文字の可視化に共通に使用されるフォントデータが、たとえば固定ディスク、フロッピーまたは集積回路ボードなどの外部ファイル22に格納されていることである。つまり、これらのフォントデータは、表示装置18やプリンタ20に依存せず、データベースとして共通に利用できる。勿論、外部ファイル22は、ホスト10に固有の記憶装置のみならず、たとえば通信回線を通して遠方のデータベースにアクセスできるものでもよい。したがって、一方ではフォントがRIP処理系に依存せず、たとえばローマンから漢字までの広範囲な拡張性があるとともに、他方ではRIP処理系が文字コード体系から独立される。

RIP処理モジュール12は、後に詳述するように、RIP処理システムを含む。この点も本実施例の重要な特徴のひとつである。ホスト10は、様々なパブリッシング用アプリケーションソフト50を実行することができる。したがってRIP処理モジュール12には、それらのアプリケーションソフト50に応じて様々な種類のコマンドデータストリーム24が入力される。また、外部ファイル22のフォントデータは、RIP処理モジュール12での必要に応じてダイナミックに読み出され、RIP処理モジュール12にロードされる。RIP処理モジュール12は、コマンドデータストリーム24にその種類に応じた適切なラスタイメージ処理を施し、ラスタイメージデータ26としてホスト10またはプリンタ20へ出力する。このデータ26は、データ圧縮された形であっても、または圧縮されない形であってもよい。

RIP処理モジュール12は、その特定の構成を第２図に示すように、基本的にはRIP処理52とフォント管理54の２つの機能部を含む。アプリケーションソフト50はソフトインタフェース56でRIP処理52とインタフェースされる。

RIP処理52は、特定のアプリケーションソフト50で指定された文字のフォントデータを取り出すのに必要なフォントデータ指定情報を作成する。このフォントデータ

指定情報には、本実施例では、フォント供給者の指定に基づく書体、文字コード体系、文字コード、文字サイズなどが含まれる。書体では、たとえば漢字であれば明朝体やゴシックなどの別、文字コード体系では、日本語であればJIS83やシフトJISの別などが指定される。

RIP処理52とフォント管理54との間のインタフェース58は、フォントに依存しない共通化されたインタフェースをとる。インタフェース58は、上述のフォントデータ指定情報、およびフォントデータを含む。フォントデータは、要求に応じて外部ファイル22のデータベースからモジュール12内の領域60にロードされる。したがって、フォント管理54におけるフォントアクセスパスは、文字コード体系に応じて最適なパスをとることができる。フォント管理54は、フォントデータ60をそのままフォントデータの形で、または文字サイズデータで要求されるサイズまで拡大もしくは縮小するなどのフォント処理を行なって、フォントデータをRIP処理52へ出力する。

フォントデータの記憶領域60は本実施例では、第３図に示すように、輪郭文字キャッシュ100およびラスタ化文字キャッシュ102の２つの高速記憶領域を含む。輪郭文字キャッシュ100は、フォントデータベース22から読み込まれた文字フォントの輪郭を表わすデータすなわち輪郭データが一時蓄積される領域である。この輪郭データは、基本的にはサイズフリーであり、後に所望の大きさのラスタ化された文字データに変換されて使用される。

ラスタ化文字キャッシュ102は、輪郭文字キャッシュ100に格納された輪郭データが所望の大きさのイメージにラスタ化された結果の文字データが蓄積される記憶領域である。ラスタ化文字キャッシュ102に格納されるイメージデータは、所望の大きさに変倍されたものである。なおラスタ化は、後述のようにRIP処理52で行なわれる。このように本実施例では、フォントデータベース22から読み込まれた輪郭データは輪郭文字キャッシュ100に格納され、次にラスタ化された文字データはラスタ化文字キャッシュ102に格納されるように、文字キャッシュが２つのレベルに階層化されている。しかし、ラスタ化文字キャッシュ102は必ずしも設けられなくてもよく、文字キャッシュについては輪郭文字キャッシュ100のみを含むシステム構成をとってもよい。

輪郭文字キャッシュ100はさらに、第４図に示すように、本実施例では２つの領域、すなわち常駐領域104および非常駐領域106を含む。常駐領域104は、たとえば日本語システムではひらがなのように相対的に使用頻度の高い文字フォントの輪郭データを蓄積する記憶領域である。使用頻度の高い文字の輪郭データは、たとえばシステムの初期設定の際、フォントデータベース22から常駐領域104に読み込まれる。常駐領域104に読み込まれた輪郭データは、常にこの領域104に保持され、使用される。

非常駐領域106は、相対的に使用頻度の低い文字フォントの輪郭データを蓄積する記憶領域である。たとえば漢字の文字データは、特定の漢字が初めてフォントデータベース22にアクセスされたときにこの領域106に蓄積される。非常駐領域106の記憶容量は有限である。そこで、たとえば最も最近に読み込まれたものから順に蓄積し、記憶容量いっぱいに蓄積されているときは最も古く読み込まれた文字データを消去するFIFO動作で、文字単位に読書き制御を行なう方法がある。または、非常駐領域106に格納されている輪郭データのそれぞれの使用回数を計数し、使用回数の最も少ないデータから順に消去するように構成してもよい。

フォント管理54は本実施例では、１文字単位の管理を基本とするのが望ましい。これは、従来技術では個々のアプリケーションソフトに応じたコマンドストリームで有していたフォントデータのアクセス機構を、本実施例ではフォント管理54で吸収させるためである。フォントデータをスケーリングすると、文字の輪郭を形成している線分の長さが不揃いになったり、丸みに滑らかさが欠けたりすることがある。そのような場合、長さや丸みを補正して適切な字形にするヒンティング処理もフォント管理54にて行なわれる。

RIP処理52は、フォント管理54から受けたフォントデータをラスタ化する。ラスタ化されたイメージデータは、イメージ／フレームメモリ62に蓄積され、ページ単位の画像データに組み立てられる。イメージ／フレームメモリ62は、たとえばページ単位でラスタイメージデータを一時蓄積する記憶領域である。

イメージ／フレームメモリ62は、ビデオインタフェース64を介してプリンタ20および表示装置18とインタフェースされる。また、圧縮データ形式でインタフェースされるプリンタ20aやファクシミリ66などの出力装置に対しては、画像圧縮68によってインタフェースされる。ホスト10に対してRIP処理モジュール12はバスで接続され、このバス接続はハードウエアバスインタフェース70を介して行なわれる。

動作を説明する。まず、装置に電源を投入すると、RIP処理52はフォントデータベース22にアクセスして、相対的に高い使用頻度の文字フォント、たとえば仮名の輪郭文字データを輪郭文字キャッシュ100に読み込み、これらをその常駐領域104に蓄積し、後の使用に備える。

文字および画像データを含むコマンドデータは、アプリケーションソフト50によって作成され、ページ記述言語で記述される中間ストリームに変換される。この中間ストリームは、ソフトウエアインタフェース56を介してRIP処理52に入力される。RIP処理52では、このコマンドストリームに含まれる文字のフォントデータにアクセスするのに必要なフォントデータ指定情報を作成し、これをフォント管理54に与える。

フォント管理54は、フォントデータ指定情報に基づいて、まず、ラスタ化文字キャッシュ102にアクセスする。必要な文字サイズのラスタ化文字データがキャッシュ102に存在すれば、フォント管理54はそれを取り込む。必要な文字サイズのラスタ化文字データがキャッシュ102になければ、フォント管理54は輪郭文字キャッシュ100にアクセスし、そのデータがキャッシュ100に存在すれば、それを取り込む。たとえばひらがなは、本実施例では常にこのキャッシュ102から輪郭データの形で入手される。必要な輪郭データが両キャッシュ100および102のいずれにも存在しないときは、フォント管理54は、外部ファイル22のフォントデータベースにアクセスし、必要なフォントデータを取り込むとともに、輪郭文字キャッシュ100にも蓄積して後の利用に備える。たとえば漢字など、比較的使用頻度の低い文字フォントについては、その輪郭データは輪郭文字キャッシュ100の非常駐領域106に読み込まれる。

フォント管理54は、輪郭文字キャッシュ100のフォントデータをそのままフォントデータの形で、または必要なサイズまで拡大もしくは縮小し、RIP処理52へ出力する。仮名などの使用頻度の高い輪郭データは常駐領域104から、その他の輪郭データは非常駐領域106から読み出される。

RIP処理52では、フォント管理54から受けたフォントデータをラスタ化し、文書データとして一旦、イメージ／フレームメモリ62に蓄積する。その際、ラスタ化された文字フォントは、フォントデータ記憶領域60のラスタ化文字キャッシュ102にも格納し、以降の使用に備える。こうしてイメージ／フレームメモリ62には、ページ単位の画像データが組み立てられる。

イメージ／フレームメモリ62に蓄積されたラスタイメージは、プリンタ20または表示装置18に出力するときは、ビデオインタフェース64を介して出力され、プリンタ20a,またはファクシミリ66などの通信装置に出力するときは、画像圧縮68によって符号化圧縮される。こうしてアプリケーションソフト50からのコマンドデータストリームはRIP処理モジュール12によってラスタイメージに変換され、たとえばプリンタ20や表示装置18などのイメージ出力装置にページの形で出力される。

本実施例ではこのように、輪郭文字キャッシュ100が設けられ、フォントデータはラスタ化される前の輪郭文字データの形でこれに蓄積される。この輪郭文字データは、後に使用の必要の都度アクセスされ、所望の大きさに変倍されてラスタ化され、イメージデータとして出力される。したがって、従来のラスタ化キャッシュのみが設けられた方式に比較して、少ない文字キャッシュ記憶容量で、あらゆる文字サイズに適応できる。本実施例ではまた、ラスタ化文字キャッシュ102とともに輪郭文字キャッシュ100が階層構造をとっているので、効率のよい文字キャッシュが実現される。

（発明の効果）

11

本発明によればこのように、文字フォントの輪郭データを一時記憶するキャッシュが設けられ、フォントデータはラスタ化される前の形でこれに蓄積される。キャッシュに蓄積された輪郭データは、後に使用の必要の都度アクセスされ、所望の大きさに変倍されてラスタ化され、イメージデータに形成される。したがって、少ない文字キャッシュ記憶容量で、あらゆる文字フォントに適応できる。このような輪郭文字キャッシュがラスタ化文字キャッシュとともに階層メモリ構造をとっている場合は、さらに効率のよい文字キャッシュ制御方式が実現される。本発明ではさらに、RIP処理とフォント管理が構成上、分離され、両者の間のインタフェースはフォントに依存しない共通の形式をとることができる。

【図面の簡単な説明】

第１図は本発明によるフォントフリーなラスタイメージ処理方式の実施例の全体構成を示す機能ブロック図、

第２図は、第１図に示す実施例におけるRIP処理モジュールの構成例を示す機能ブロック図、

＊

12

＊第３図は同実施例における文字キャッシュの構成例を示す概念図、

第４図は同実施例における輪郭文字キャッシュの構成例を示す概念図である。

主要部分の符号の説明

10……ホスト

12……RIP処理モジュール

18……表示装置

22……フォントデータベース

52……RIP処理

54……フォント管理

60……フォントデータ記憶領域

62……イメージ／フレームメモリ

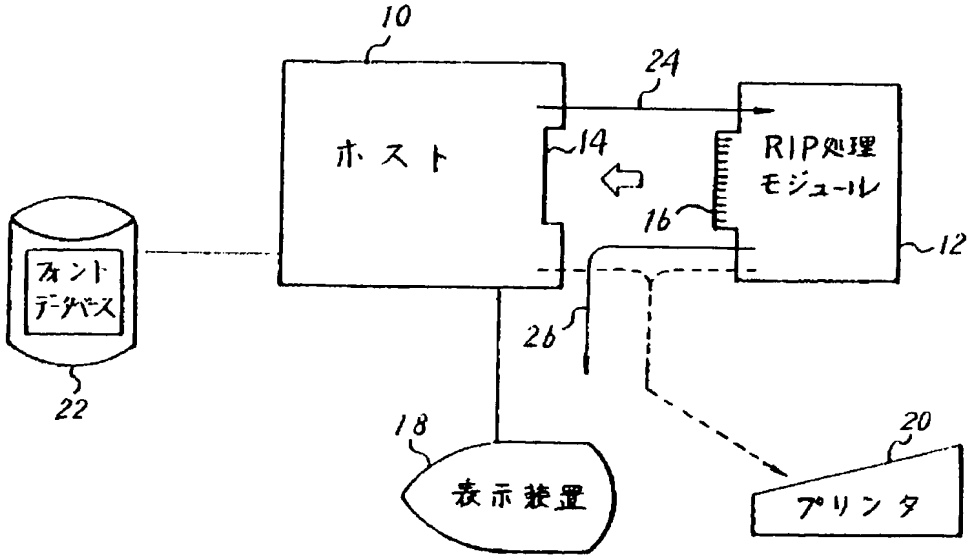
100……輪郭文字キャッシュ

102……ラスタ化文字キャッシュ

104……常駐領域

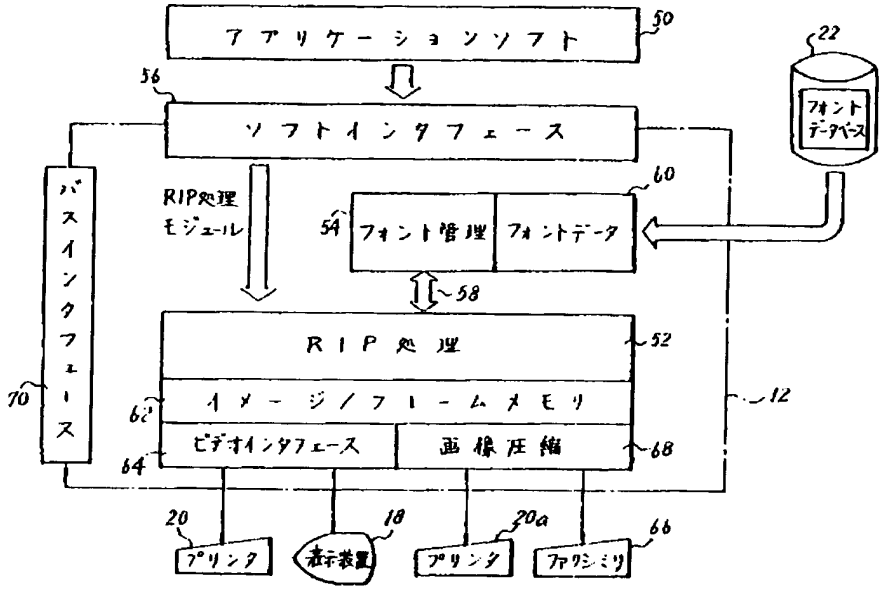
106……非常駐領域

【第１図】



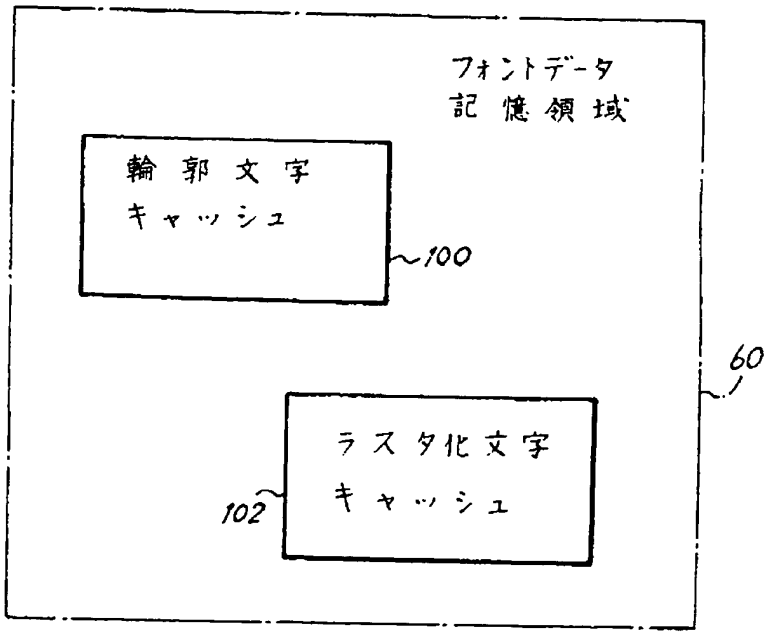
フォントフリーなラスタイメージ処理システム

【第2図】



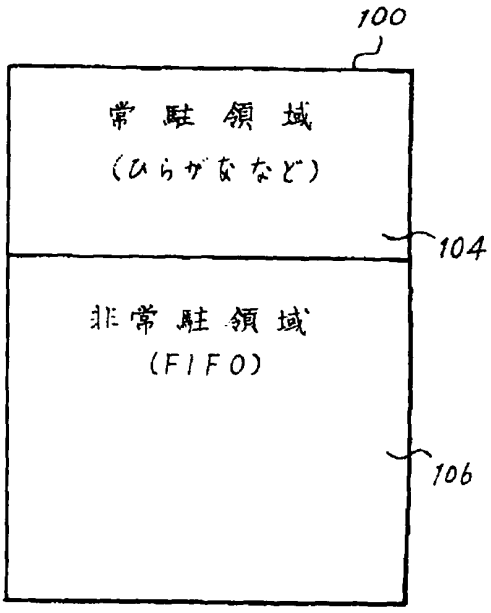
RIP処理モジュールの構成例

【第3図】



文字キャッシュの構成例

【第４図】



輪郭文字キャッシュの構成例

フロントページの続き

(72)発明者　小山　法孝
東京都港区虎ノ門１丁目７番12号　沖電気工業株式会社内

(56)参考文献　特開　平１－275057（ＪＰ，Ａ）
特開　平１－188354（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[6]，ＤＢ名)
G09G　5/22